METHOD AND SYSTEM FOR CONGESTION AVOIDANCE IN PACKET SWITCHING DEVICES

FIELD OF THE INVENTION

The present invention relates to congestion avoidance in packet switching devices and, more particularly, to congestion avoidance using packet discard techniques.

BACKGROUND OF THE INVENTION

The aggregate link bandwidth for all links supported by a packet switching device (e.g., a router) is often higher than the total switching capacity of the device. This causes congestion at buffers located at the ingress (pre-switching) and egress (post-switching) stages of the device. Congestion may even arise in switch fabrics that are non-blocking. Commonly, buffering may be used to control congestion. However, buffering may cause delays in the delivery of packets and therefore in order to avoid excessive delays, a more sophisticated technique for congestion needs to be developed.

One solution for controlling congestion is a tail drop approach, whereby an egress buffer is allowed to fill and then packets are dropped if they arrive from the switch fabric while the buffer is full. However, this approach may cause multiple flows to suffer lost packets. The higher layer protocols may react to this type of packet loss by terminating the flow and re-transmitting the lost information. Although congestion per se has been

eliminated, the end result will be a highly undesirable slowdown-speedup-slowdown-etc. behaviour of the packet switching device.

In another conventional congestion control algorithms, the egress stage takes an action on each packet, such action being either to queue the packet or discard it. An example of an algorithm of this type is a random early discard (RED) algorithm. Specific examples of RED algorithms include the RED algorithm (described in Internet Request For Comments (RFC) 2309, April 1998, incorporated by reference herein) and the BLUE algorithm (described in "BLUE: A New Class of Active Queue Management Algorithms", Wu-chang Feng et al., pages 1-26, incorporated by reference herein). The decision as to whether a packet should be discarded or queued is made by monitoring the degree to which the egress buffer on a given link is full and consequently generating a discard probability for that packet. If a random number generated for that packet is below the discard probability, the packet is discarded; otherwise it is placed in the egress buffer. In this way, congestion at the egress buffer can be controlled by actions taken at the egress buffer.

However, adding to jitter and latency by delaying packets that will not be discarded and sending packets that will be discarded requires switch fabrics to be significantly over-provisioned. Thus, by the very action of discarding or queuing a packet at the device egress (i.e., once switching resources have already been utilized to switch the packet), those packets that are eventually discarded

will have unnecessarily consumed resources throughout the ingress and switching stages of the packet switching device. Clearly, by making decisions based on measured congestion levels, there will inevitably be an inefficiency regarding the extent to which the memory and/or switching resources of the device are utilized.

Accordingly, there is a need in the industry to develop a mechanism that limits congestion while resulting in more efficient resource utilization within a packet switching device such as a router.

SUMMARY OF THE INVENTION

The present invention provides a method for regulating packet flow at the ingress stage of a packet switching device. Specifically, bandwidth utilization information is obtained for each of a plurality of queues at the egress stage of the device. Based on the bandwidth utilization information, computations are performed to evaluate a "discard probability" for each queue. This information is transmitted to the ingress stage, either periodically or at other controlled time periods, such as when the discard probability changes significantly. The ingress stage can then proceed with controllable transmission or non-transmission of packets to the switch fabric, depending on the queue for which the packet is destined and also depending on the discard probability for that queue. In this way, congestion can be avoided even before it even has a chance to occur. This leads to improved bandwidth utilization, since packets which are

discarded at the ingress stage will not unnecessarily take up other resources in the device.

Accordingly, the present invention may be summarized as a method of regulating packet flow through a device having a switch fabric with a plurality of input ports and a plurality of output ports, a control entity connected to the input ports for regulating packet flow thereto, and a plurality of egress queues connected to the output ports for temporarily storing packets received therefrom. The method includes obtaining bandwidth utilization information regarding packets received at the egress queues; determining, from the bandwidth utilization information, a discard probability associated with each egress queue; and providing the discard probability associated with each egress queue to the control entity, for use by the control entity in selectively transmitting packets to the input ports of the switch fabric.

In a specific embodiment, obtaining bandwidth utilization information regarding packets received at the egress queues may include determining, for each particular one of the output ports, an average idle time between successive packets received from the particular output port; determining for each particular one of the output ports, an average number of traffic bytes received per time unit for each egress queue connected to the particular output port and determining, for each particular one of the output ports, an average number of non-traffic bytes received per time unit from the particular output port.

In a specific embodiment, a discard probability for a particular one of the egress queues may then be determined by determining an allocated traffic bandwidth for the particular egress queue and comparing the average number of received traffic bytes for the particular egress queue to the allocated traffic bandwidth for the particular egress queue. If the average number of received traffic bytes for the particular egress queue is greater (less) than the allocated traffic bandwidth for the particular egress queue, the discard probability for the particular egress queue is set to the sum of a time average of previous values of the discard probability for the particular egress queue and a positive (negative) increment.

In a specific embodiment, a discard probability could be computed for each combination of egress queue and packet priority.

In a specific embodiment, the method of the invention may be embodied as a sequence of instructions on a computer-readable storage medium.

The method may be summarized according to a second broad aspect as a drop probability evaluation module, which includes an allocation processing entity, for determining an allocated traffic bandwidth for each of the egress queues and a probability processing entity in communication with the allocation processing entity, the probability processing entity being adapted to receive the allocated traffic bandwidth for each of the egress queues from the allocation processing entity and also

adapted to receive an average number of received traffic bytes for each of the egress queues from an external entity.

The probability processing entity is operable to compare the average number of received traffic bytes for each particular one of the egress queues to the allocated traffic bandwidth for the particular egress queue and set the discard probability for the particular egress queue to the sum of a time average of previous values of the discard probability for the particular egress queue and either a positive or a negative increment, depending on whether the average number of received traffic bytes for the particular egress queue is greater or less than the allocated traffic bandwidth for the particular egress queue.

According to a third broad aspect, the present invention may be summarized as a device equipped with a switch fabric having a plurality of input ports and a plurality of output ports, the switch fabric being adapted to switch packets between its input ports and its output ports. The device also includes a plurality of egress queues connected to corresponding ones of the output ports of the switch fabric, each egress queue being adapted to (i) temporarily store packets received from the corresponding output port of the switch fabric and (ii) determine bandwidth utilization information on the basis of the packets received at the egress queues.

The device further includes a drop probability evaluation module connected to the egress queues, the drop

probability evaluation entity being adapted to determine a discard probability associated with each of the egress queues on the basis of the bandwidth utilization information. The device also includes a packet acceptance unit connected to the input ports of the switch fabric and to the drop probability evaluation module, the packet acceptance entity being adapted to (i) receive packets destined for the output ports of the switch fabric; (ii) identify an egress queue associated with each received packet; and (iii) on the basis of the discard probability associated with the egress queue associated with each received packet, either transmit or not transmit the received packet to one of the input ports of the switch fabric.

According to still another broad aspect, the present invention may be summarized as a method of regulating packet flow through a device having an ingress entity, an egress entity, a processing fabric between the ingress entity and the egress entity, and a control entity adapted to process packets prior to transmission thereof to the ingress entity. The method includes obtaining congestion information regarding packets received at the egress entity and providing the congestion information to the control entity, for use by the control entity in processing packets prior to transmission thereof to the ingress entity.

These and other aspects and features of the present invention will now become apparent to those of ordinary skill in the art upon review of the following description

of specific embodiments of the invention in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

In the drawings:

Fig. 1 is a block diagram of a packet switching device equipped with random packet discard functionality in accordance with an embodiment of the present invention; and

Fig. 2 is a block diagram of an embodiment of a discard probability evaluation module in the device of Fig. 1.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

With reference to Fig. 1, there is shown a packet switching device in accordance with an embodiment of the present invention, which implements congestion avoidance by either transmitting or not transmitting packets at an ingress stage, based on congestion information obtained at an egress stage.

In one embodiment, the packet switching device 100 is a multi-stage router and the egress stage may be an intermediate or final stage in the multi-stage router. The device 100, which can be connected to adjacent devices (e.g., routers or nodes) in a packet switched network, includes a switch fabric 102 with a plurality of input ports 104 and a plurality of output ports 106. The input ports 104 of the switch fabric 102 are connected to

a set of input line cards 108 and the output ports 106 of the switch fabric 102 are connected to a set of output line cards 110. In embodiments where the line cards are bi-directional, it is possible that there will be no distinction between the input and output line cards 108, 110. Also, in a multi-stage router, the output line cards 110 would be embodied as a next stage of routing.

The input line cards 108 are adapted to receive streams of packets from an external entity and provide them to the input ports 104 of the switch fabric 102. Each input line card 108 may be connected to one, two or more input ports 104 of the switch fabric 102 via a bus architecture 105. The output line cards 110 are adapted to receive streams of packets from the output ports 106 of the switch fabric 102 and to forward them further downstream to an adjacent router or node of the network. Each output line card 110 has one, two or more physical interfaces, which correspond to individual ones of the output ports 106 of the switch fabric 102. The interfaces on each individual output line card 110 are connected to the corresponding output ports 106 of the switch fabric 102 via a bus architecture 107 common to that output line card 110. In the illustrated embodiment, there are four interfaces denoted I_0, I_1, I_2, I_3, with two interfaces being located on each of two output line cards 110. However, it should be understood that other arrangements are possible and that the present invention is not limited to any particular number of interfaces, output line cards 110 or distribution of interfaces amongst the output line cards 110.

A packet is typically formed by a header and a payload, and may be associated with a priority (or precedence or service class). The header usually identifies the priority of the packet (if applicable) as well as a destination node for which the packet is destined. The input line card 108 that receives a particular packet translates its destination node into a destination output port, say 106_X, of the switch fabric 102 and inserts the identity of this destination output port 106_X into the header of the packet. The packet, with the identity of destination output port 106_X now specified in the header, is then provided via the appropriate bus 105 to an input port 104 of the switch fabric 102. The switch fabric 102 is responsible for ensuring that the packet indeed emerges at the desired destination output port 106_X. The output line card 110 that has an interface connected to destination output port 106_X (via the appropriate bus 107 for that output line card) removes the identity of the destination output port 106_X from the header of the packet and forwards the packet, in its original form, to an adjacent router or node of the network. In this way, the packet is moved closer to its destination node. In the case of a multi-stage router, the output line card would represent a next stage of routing.

In one embodiment, the switch fabric 102 is capable of supplying information to each output line card 110 at a higher rate than the one at which the output line card 110 can transmit out of the device 100. This is in order to allow the output line card 110 to buffer the excess bandwidth and make intelligent decisions about how to route the buffered packets. In other embodiments, the

egress entity need not be a queue, strictly speaking, if the rate of departure of packets from the device 100 matches or is less than the rate that can be handled by an external device connected to the output line cards 110.

Regardless of whether a buffering capability is required at the egress of the device 100, the interfaces on the output line cards 110 contain an egress entity. In one embodiment, each interface on each of the output line cards 110 is associated with an one, two or more egress queues 112 forming part of the egress entity. The egress queues are used to temporarily store packets in the stream of packets arriving from the corresponding output port 106 of the switch fabric 102 via the appropriate bus 107. The egress queues 112 can be said to behave as virtual interfaces or virtual ports for the physical interface with which they are associated. It should be mentioned that in a multi-stage router, the egress queues 112 may in fact be ingress queues with respect to a subsequent stage of routing.

In the illustrated embodiment, there are two egress queues 112 per interface and are individually denoted Q_0, Q_1. Thus, interface I_0 is associated with its own egress queues Q_0, Q_1, interface I_1 is associated with its own egress queues Q_0, Q_1, etc. However, it should be understood that other arrangements are possible and that the present invention is not limited to any particular number of egress queues 112 per interface. Moreover, in the case where packets can have different priorities (e.g., P_0 and P_1), the egress queues 112 may each be

divided into a respective set of sub-queues 113 on the basis of priority. It should be appreciated that the sub-queues 113 need not all be of the same depth, and there need not be the same number of sub-queues 113 for each egress queue 112.

According to an embodiment of the present invention, there is also provided an egress traffic manager (ETM) 114 for each interface. Each egress traffic manager 114 comprises suitable circuitry, software and/or control logic for removing the identity of the destination output from the header of each received packet and also for determining to which interface and into which egress queue 112 and sub-queue 113 the received packet is to be placed. It should be appreciated that in other embodiments of the present invention, there may be provided one egress traffic manager 114 per output line card 110, such that each egress traffic manager 114 would be connected directly in the path of a corresponding one of the buses 107.

The determination of the interface to which to transmit a received packet may be made on the basis of information (such as the destination node) specified in the received packet's header. The determination of the egress queue 112 into which to insert a received packet may be made on the basis of information (such as a virtual port identifier) specified in the received packet's header. The determination of the sub-queue 113 into which to insert a received packet may be made on the basis of information (such as the priority) specified in the received packet's header.

Assuming for the purposes of illustration that there is in fact one egress traffic manager 114 per interface, each such egress traffic manager 114 is additionally equipped with circuitry, software and/or control logic for monitoring the number and destination of packets received from the corresponding output port 106 of the switch fabric 102. On the basis of this information, the egress traffic manager 114 generates information indicative of congestion.

The congestion information may include bandwidth utilization information, such as:

an average idle time between received packets at interface I_i (denoted AIT(I_i) - **A**verage **I**dle **T**ime);

an average received non-traffic bytes value for interface I_i (denoted ARNB(I_i) - **A**verage **R**eceived **N**on-traffic **B**ytes); and

an average received traffic bytes value for each priority $P \in \{113_0, 113_1\}$ of each egress queue $Q \in \{112_0, 112_1\}$ associated with interface I_i (denoted ARTB(I_i,Q,P) - **A**verage **R**eceived **T**raffic **B**ytes).

The congestion information may alternatively include a measure of the depth of each egress queue 112 or a measure of the variability of each queue. Assuming for the purposes of illustration that the congestion information is bandwidth utilization information produced by each egress traffic manager 114 located on a given

output line card 110, such information is provided to a common discard probability evaluation module (DPEM) 120 for that output line card 110. (Alternatively, there may be a separate DPEM 120 for each egress traffic manager 114 on the output line card 110.)

The DPEM 120 on a given output line card 110 comprises circuitry, software and/or control logic for computing a discard probability for each egress queue 112 and sub-queue 113 associated with each interface on that given output line card 110. Thus, each DPEM 120 will be responsible for computing the discard probabilities for one, two or more interfaces, depending on the number of interfaces on the output line card where the DPEM 120 is located. For notational convenience, the discard probability for interface I_i, queue 112_q and sub-queue 113_p shall be denoted $DP(I_i,112_q,113_p)$.

Each DPEM 120 is connected via a control link 122 to one or more packet acceptance units (PAUs) 118 in each input line card 108. The control link from a given DPEM 120 to the input line cards 108 carries the discard probability $DP(I,Q,P)$ for each combination of queue and priority that are possible for each interface associated with the output line card 110 containing the DPEM 120. Since this is done by each DPEM 120 in each output line card 110, each PAU 118 in each input line card 108 will therefore have access to the discard probability for every possible combination of interface, queue and sub-queue. The discard probabilities transmitted by a particular DPEM 120 may be sent in the form of a broadcast message. The switch fabric 102 may in fact be used as a channel for

carrying the discard probabilities $DP(I,Q,P)$ from each DPEM 120 to the PAUs 118.

Considering the PAU 118 in a given one of the input line cards 108, this unit is employed for processing a stream of packets prior to the packets' transmission to a corresponding input port 104 of the switch fabric 102. One of the functions of the PAU 118 is to implement congestion avoidance functionality using random discard of packets, based upon the interface and egress queue (and priority, if applicable) of each packet and based upon the discard probability associated with that combination of interface, egress queue (and priority, if applicable). Although the illustrated embodiment shows one PAU 118 per input port 104 of the switch fabric 102, in some embodiments it may be advantageous to provide one PAU 118 per input line card 108 or a single PAU 118 that is distributed amongst the input line cards 108.

The PAU 118 assigned to process a particular stream of packets is equipped with suitable circuitry, software and/or control logic for determining the destination output port of each received packet. This destination output port will correspond to one of the interfaces (say, I_1) on one of the output line cards 110. In addition, the PAU 118 comprises suitable circuitry, software and/or control logic for determining one of the egress queues 112 (say, 112_q) into which the received packet will be inserted by the egress traffic manager 114 associated with the interface corresponding to the destination output port. Moreover, if a received packet can have either of several priorities, then the PAU 118

further includes suitable circuitry, software and/or control logic for determining the priority associated with the packet and hence the sub-queue (say, 113_p) into which the packet is to be inserted. Based on this information and also based on the discard probability $DP(I_i,112_q,113_p)$, the PAU 118 makes a decision as to whether it should drop the received packet or continue with its transmission towards the switch fabric 102.

In order to make its decision as to whether or not to drop a received packet characterized by interface I_i, egress queue 112_q and sub-queue 113_p, the PAU 118 includes suitable circuitry, software and/or control logic for generating a random number R for the received packet and for comparing R to the discard probability $DP(I_i,112_q,113_p)$. If R is, say, lower than $DP(I_i,112_q,113_p)$, then the packet is discarded, otherwise the packet is sent into the corresponding input port 104 of the switch fabric 102. Alternatively, the packet can be discarded if the random number R is higher than discard probability $DP(I_i,112_q,113_p)$. The term "random number" in this sense is meant to include a number generated by pseudo-random or quasi-random techniques.

In the case where it is decided that a received packet is indeed to be forwarded to the switch fabric 102, the PAU 118 comprises suitable circuitry, software and/or control logic for inserting the identity of the destination output port into the header of the packet and to forward the packet, whose header now specifies the identity of the destination output port, to the corresponding input port 104 of the switch fabric 102. However, in the case

where it is decided that the received packet is to be discarded, the packet is not transmitted to the switch fabric 102 and may be discarded from memory altogether or sent to a separate memory store for discarded packets. Advantageously, packets that are not transmitted do not consume resources in the switch fabric 102 or in the congestion management facility of the PAU 118, leading to improved resource utilization.

Generation of the bandwidth utilization values (i.e., AIT(I_0), ARBN(I_0), and ARBT(I_0,Q,P)) by the egress traffic manager 114 associated with interface I_0 is now described. Firstly, with respect to the AIT(I_0) value, this is an indication of overall bandwidth utilization of interface I_0. If fixed-length packets are used, then overall bandwidth utilization could be measured directly by counting the number of packet arrivals per second at the egress traffic manager 114. In such a case, computation of the average idle time is not necessary. However, if variable-length packets are used, overall bandwidth utilization is preferably measured indirectly, e.g., by evaluating the average duration of the interval of non-transmission between successive received packets. This is referred to the average idle time between packets.

Implementation of an approach whereby the average idle time between packets is measured is facilitated if a dedicated bit in a word is used to indicate whether that word is a certain number of words away from the last word in the packet to which that word belongs. Such a technique for signaling the imminent end of a packet is described in United States patent application serial no.

09/870,766 to Norman et al., filed on May 31, 2001, assigned to the assignee of the present invention and hereby incorporated by reference herein.

The egress traffic manager 114 associated with interface I_0 also generates the ARTB(I_0,Q,P) values for $Q \in \{112_0, 112_1\}$, $P \in \{113_0, 113_1\}$, which is indicative of the average number of traffic bytes destined for each combination of egress queue and sub-queue for interface I_0. A traffic byte is a byte belonging to a packet that must meet certain user-oriented quality of service criteria. In other words, traffic bytes belong to packets for which congestion avoidance is to be performed. In order to compute each ARTB(I_0,Q,P) value, the egress traffic manager 114 comprises suitable circuitry, software and/or control logic for analyzing the header of each incoming packet and, from the information in the header, determining the egress queue 112 for which the packet is destined, as well as the priority of the packet.

Additionally, the egress traffic manager 114 associated with interface I_0 also generates the ARNB(I_0) value for, which is indicative of the average number of non-traffic bytes received at interface I_0. A non-traffic byte belongs to a packet to which user-oriented quality of service criteria are not attached. In order to compute the ARNB(I_0) value, the egress traffic manager 114 comprises suitable circuitry, software and/or control logic for analyzing the header of each incoming packet and, from the information in the header, determining whether the packet is a traffic packet or a non-traffic

packet. It should be understood that the analysis of each packet's header may be done only once for each packet, in the context of computing both the ARNB(I_0) value and the ARTB(I_0,Q,P) value.

An example of a discard probability evaluation module (DPEM) 120 suitable for computation of the discard probability DP(I_i,112_q,113_p) for each valid combination of i, q and p is now described in greater detail with reference to Fig. 2. Specifically, the illustrated DPEM 120, which is associated with one of the output line cards 110, includes an aggregation processing entity 208, an availability processing entity 210, an allocation processing entity 220, a probability processing entity 230 and an extrapolation processing entity 240.

The aggregation processing entity 208 receives the ARNB(I_0) value and the AIT(I_0) value from the egress traffic manager 114 associated with interface I_0, and the ARNB(I_1) value and the AIT(I_0) value from the egress traffic manager 114 associated with interface I_1. Based on its inputs, the aggregation processing entity 208 determines an aggregate average number of received non-traffic bytes (denoted ARNB), as well as a bandwidth gradient (denoted BWGR). The ARNB and BWGR values are provided to the availability processing entity 210. Computation of the ARNB value can be done by adding the ARNB(I_1) values are added for i = 0 and i = 1. Computation of the BWGR value can be done as follows:

The measured average idle time AIT(I_0), AIT(I_1) for each interface is averaged, in order to come up with an

aggregate average idle time. The aggregate average idle time is then compared to a set of pre-determined thresholds. In one embodiment, the aggregate average idle time for each interface is first compared to a critical minimum average threshold (denoted T_1). If it is less than T_1, then this situation is indicative of a critical over-utilization of bandwidth within the switch fabric 102. The bandwidth gradient value (BWGR) is consequently set to indicate that an urgent bandwidth decrement is required at the ingress side.

If, however, the aggregate average idle time is not less than T_1, then it is compared to a pre-determined minimum average threshold (denoted T_2). If the aggregate average idle time is less than T_2, then this situation is indicative of non-critical congestion that consumes buffer space within the switch fabric 102. The bandwidth gradient value (BWGR) is consequently set to indicate that a moderate bandwidth decrement is required at the ingress side.

If the aggregate average idle time is not less than T_2, then it is compared to a pre-determined maximum average threshold (denoted T_3). If the aggregate average idle time is greater than T_3, then this situation is indicative of an under-utilization of bandwidth within the switch fabric 102. The bandwidth gradient value (BWGR) is consequently set to indicate that a bandwidth increment is required at the ingress side.

Finally, if the aggregate average idle time is between T_2 and T_3, then this situation is indicative of a utilization

of bandwidth within the switch fabric 102 which does not require compensation. The bandwidth gradient value (BWGR) is consequently set to indicate that neither a bandwidth increment nor a bandwidth decrement is required at the ingress side.

It should be noted that the thresholds T_1, T_2 and T_3 can be adjusted dynamically based on parameters such as bandwidth utilization and possibly, in addition, queue depth and bandwidth variability (burstiness).

The availability processing entity 220 receives the BWGR value (i.e., the bandwidth gradient) and the ARNB value (i.e., the average received non-traffic bytes) from the aggregation processing entity 208. Based on its inputs, the availability processing entity 210 determines a total available bandwidth for traffic packets, which is supplied to the allocation processing entity 220 in the form of a BWAVAIL (**B**and**W**idth **AVAIL**able) value. Computation of the BWAV value can be done as follows:

The availability processing entity keeps an internal record of the aggregate bandwidth available to all packets (both traffic packets and non-traffic packets), which may be denoted AGG_AVAIL. If the BWGR value is indicative of a bandwidth increment being required at the ingress side, then AGG_AVAIL is incremented by a pre-configured step value, up to a maximum aggregate available bandwidth; if the BWGR value is indicative of a bandwidth decrement being required at the ingress side, then AGG_AVAIL is decremented by a pre-configured step value, down to a minimum aggregate available bandwidth;

if the BWGR value is indicative of neither a bandwidth increment nor a bandwidth decrement being required at the ingress side, then AGG_AVAIL remains unchanged; and if the BWGR value is indicative of an urgent bandwidth decrement being required at the ingress side, then AGG_AVAIL is set to a pre-configured (low) value.

Next, the ARNB value is subtracted from the resultant value for AGG_AVAIL, yielding the BWAVAIL value, which is the total bandwidth available only for traffic packets. In one embodiment of the present invention, the step values for the increment and decrement operations may each be percentages of the minimum aggregate available bandwidth. Since it may be more important to decrement bandwidth than to increment it, the step value for the increment operation may be lower than the step value for the decrement operation.

The allocation processing entity 220, in addition to receiving the total available bandwidth for traffic packets from the availability processing entity 210 in the form of the BWAVAIL value, also receives an indication of the average number of bytes that would be received for each egress queue 112 on the output line card if the discard probability were zero. This information is received from the extrapolation processing entity 240 in the form of a plurality of ARTBDP0(I,Q) values (i.e., **A**verage **R**eceived **T**raffic **B**ytes if the **D**iscard **P**robability were **0**), where $I \in \{I_0, I_1\}$ and $Q \in \{112_0, 112_1\}$. Computation of each ARTBDP0(I,Q) value is described in greater detail later on in the context of the extrapolation processing entity 240.

Based on its inputs, the allocation processing entity 220 allocates bandwidth for traffic bytes for each valid combination of I and Q. The outcome of this computation is provided to the probability processing entity 230 in the form of an allocated bandwidth value (denoted BWALLOC(I,Q) - **B**and**W**idth **ALLOC**ated) for that combination of I and Q.

Computation of the BWALLOC(I,Q) value can be done as follows: The allocation processing entity 220 first determines whether the bandwidth commitments for each combination of I and Q are being met. This is done by comparing the previous value of BWALLOC(I,Q) to the corresponding ARTBDP0(I,Q) value. Thus, the allocated bandwidth is being compared to the maximum possible bandwidth that could be received for that combination of I and Q.

If BWALLOC(I,Q) exceeds ARTBDP0(I,Q), then BWALLOC(I,Q) is decreased, e.g., by a fixed amount or by a factor that depends on the difference between BWALLOC(I,Q) and ARTBDP0(I,Q). On the other hand, if BWALLOC(I,Q) is less than ARTBDP0(I,Q), then BWALLOC(I,Q) is increased, e.g., by a fixed amount or by a factor that depends on the difference between ARTBDP0(I,Q) and BWALLOC(I,Q). The incremented or decremented values of BWALLOC(I,Q) are supplied to the probability processing entity 230.

It should be noted that alternative embodiments, in which an outcome of "no change" could be applied to a particular BWALLOC(I,Q) values, are also within the scope

of the present invention. It should further be noted that it is advantageous to perform a check in order to ensure that the sum of BWALLOC(I,Q) over all I and Q for the same line card does not exceed BWAVAIL for that line card, as received from the availability processing entity 210.

The probability processing entity 230, in addition to receiving the BWALLOC(I,Q) values (for $I \in \{I_0, I_1\}$ and $Q \in \{112_0, 112_1\}$) from the allocation processing entity 220, also receives the ARTBDP0(I,Q,P) values (for $I \in \{I_0, I_1\}$, $Q \in \{112_0, 112_1\}$ and $P \in \{113_0, 113_1\}$) from the extrapolation processing entity 240, the ARTB(I_0,Q,P) values (for $Q \in \{112_0, 112_1\}$ and $P \in \{113_0, 113_1\}$) from the egress traffic manager 114 associated with interface I_0 and the ARTB(I_1,Q,P) values (for $Q \in \{112_0, 112_1\}$ and $P \in \{113_0, 113_1\}$)from the egress traffic manager 114 associated with interface I_1.

Based on its inputs, the probability processing entity 230 generates the discard probability DP(I,Q,P) for each valid combination of I, Q and P, in this case for $I \in \{I_0, I_1\}$, $Q \in \{112_0, 112_1\}$ and $P \in \{113_0, 113_1\}$. Computation of the discard probability DP(I,Q,P) for all values of P for a given value of I (say, i) and Q (say, q) can be done as follows:

Firstly, the sum of the ARTB(i,q,P) is taken over all P. This leads to a quantity that represents the total average number of received traffic bytes for egress queue 112_q associated with interface I_i, which may be denoted TARTB(i,q). This quantity is compared to BWALLOC(i,q),

in order to determine whether more bandwidth than is required has been allocated. Since optimal resource usage efficiency occurs when the allocated bandwidth matches the actual bandwidth used, the difference in the two quantities provides an error signal that is to be driven to zero. At this stage, it is possible to take a simple approach and a more complex approach. The simple approach will be described first, followed by the more complex approach.

In the event that the allocated bandwidth is greater than the total average bandwidth used, the discard probability DP(i,q,*P*) will, in the simple approach, be decreased for one or more *P* (depending on whether an intserv or diffserv model is applied) so that fewer packets are discarded at the ingress, resulting in an eventual increase in TARTB(i,q). Conversely, if the total average bandwidth is less than the actual bandwidth used, the discard probability DP(i,q,*P*) will be increased for one or more *P* (depending on whether an intserv or diffserv model is applied) so that a greater number of packets are discarded at the ingress, resulting in an eventual decrease in TARTB(i,q). The magnitude of an increase applied to the discard probability DP(i,q,*P*) could be different from the magnitude of a decrease.

The above procedure is repeated until the allocated bandwidth is within a certain range of the total average bandwidth used. Advantageously, this provides a certain level of congestion avoidance. However, convergence may take a relatively long time to occur. This is due to the fact that a large amount of time will elapse between a

change in the discard probability and a corresponding change in the average number of received traffic bytes. Moreover, if the discard probability is altered before a change in the average number of received traffic bytes can be detected, then it is possible to "overshoot" the final discard probability that would allow the allocated bandwidth to be within a certain range of the total average bandwidth used. In order to reduce the convergence time, one may have recourse to a more complex approach.

In the more complex approach, the net amount by which the discard probability for each *P* is increased / decreased is the result of an iterative procedure which relies on (i) a time average of the discard probability (denoted ADP(i,q,P) and is provided to the extrapolation processing entity 240); (ii) a temporary discard probability (denoted $DP_{temp}(i,q,P)$; and (iii) a temporary average number of received traffic bytes (denoted $ARTB_{temp}(i,q,P)$).

At initialization, the temporary drop probability $DP_{temp}(i,q,P)$ is set to the previous version of DP(i,q,*P*) and the temporary average number of received traffic bytes $ARTB_{temp}(i,q,P)$) is set to the previous average number of received traffic bytes ARTB (i,q,*P*)). The iterative procedure starts by determining whether an increase or decrease in the drop probability is required by comparing, as before, the allocated bandwidth BWALLOC(i,q) and the total average bandwidth used TARTB (i,q). Depending on whether an increase or decrease is

required, the value of the *temporary* drop probability for one or more *P* is changed accordingly.

At this point, the temporary average number of received traffic bytes $ARTB_{temp}(i,q,P)$ is altered, but in the opposite direction. Thus, an increase in the temporary drop probability corresponds to a decrease in the temporary average number of received bytes, while a decrease in the temporary drop probability corresponds to an increase in the temporary average number of received bytes. With the new value for each temporary average number of received traffic bytes, the total temporary average bandwidth used $TARTB_{temp}(i,q)$ is computed by summing together the values of $ARTB_{temp}(i,q,P)$ for all *P*. The value of $TARTB_{temp}(i,q)$ is compared to $BWALLOC(i,q)$ and the result will be indicative of whether the allocated bandwidth is greater than the *expected* total average bandwidth used.

The steps of changing the temporary drop probability $DP_{temp}(i,q,P)$ for one or more *P* and re-evaluating the values of $ARTB_{temp}(i,q,P)$ for all *P* and the value of $TARTB_{temp}(i,q)$ can be repeated many times. In one embodiment, the steps are repeated until the value of $TARTB_{temp}(i,q)$ is to within a desired range of $BWALLOC(i,q)$. Alternatively, the steps may be repeated a fixed number of times or until convergence of the temporary drop probability is reached. In any event, after the required amount of iterations, each drop probability $DP(i,q,P)$ is set to the current value of the corresponding temporary drop probability $DP_{temp}(i,q,P)$ and is provided to the PAUs 118 in the input line cards. In

this way, it is possible to predict the changes in bandwidth utilization resulting from a change in discard probability in order to arrive at the desired bandwidth utilization more quickly.

Different initial step sizes may be used for the $DP_{temp}(i,q,P)$ and $ARTB_{temp}(i,q,P)$ values. For the $DP_{temp}(i,q,P)$ values, the initial step size may be a fixed value. For the $ARTB_{temp}(i,q,P)$ values, the initial step size may be a value that depends on the , which is then recued by powers of two at each (or every N^{th}) iteration Also, it is advantageous at each (or every N^{th}) iteration to reduce the step size for increasing or decreasing the $DP_{temp}(i,q,P)$ values and the $ARTB_{temp}(i,q,P)$ values with respect to their previous values. By way of a non-limiting example, the reduction may be logarithmic (e.g., by a power of two).

It should be appreciated that those skilled in the art may be motivated to improve the performance of the probability processing entity 230 by evaluating the discard probabilities not only as a function of bandwidth utilization, but also as a function of other parameters, such as the depth of the egress queues 112 and/or sub-queues and the bandwidth variability (burstiness) of the individual streams flowing to each interface, egress queue and/or sub-queue. The burstiness of a stream can be viewed as the derivative of the bandwidth of that stream.

The extrapolation processing entity 240, in addition to receiving the $ADP(I,Q,P)$ values from the probability

processing entity 230, also receives the ARTB(I,Q,P) values, i.e., the average received traffic bytes for each valid combination of I, Q and P, from the egress traffic manager 114. Based on its inputs, the extrapolation processing entity 240 computes the average number of received traffic bytes if the discard probability were zero, for each valid combination of I, Q and P. These values are supplied to the probability processing entity 230 in the form of the ARTBDP0(I,Q,P) values.

Computation of the ARTBDP0(I,Q,P) values can be done as follows: Given the average number of received traffic bytes for each valid combination of I, Q and P, and given the average discard probability for the same combination of I, Q and P, an extrapolation can be performed to calculate what the average number of received traffic bytes would be if the discard probability were zero. Specifically, ARTBDP0(I,Q,P) = ARTB(I,Q,P) / (1-ADP(I,Q,P)). If the ARTB(I,Q,P) values are received more often than the ADP(I,Q,P) values, then upon receipt of an ARTB(I,Q,P) value, one would read the corresponding ADP(I,Q,P) value and compute the corresponding ARTBDP0(I,Q,P) value.

Additionally, the extrapolation processing entity 240 also performs a summation of the ARTBDP0(I,Q,P) values over all priorities associated with a common interface and egress queue, which yields the average number of received bytes for a given combination of I and Q worst-case received bytes for that combination of I and Q. The extrapolation processing entity 240 supplies this information to the allocation processing entity 220 in

the form of the ARTBDP0(I,Q) value for that combination of I and Q. Thus, ARTBDP0(I,Q) = $\sum$(ARTBDP0(I,Q,P))$_P$, where $\sum(x)_P$ denotes summation of the set of $x(P)$ over all P.

In some embodiments, it may be advantageous to limit the rate at which the DP(I,Q,P) values are transmitted to the PAUs 118 in order to limit the flow of non-traffic bytes through the switch fabric 102 and also to limit instabilities due to long reaction times following a change in the discard probabilities. Instead of refreshing at a high rate, a new discard probability for a given (I,Q,P) triplet may be sent whenever it has changed from its previous value by more than a pre-determined absolute or relative amount. This pre-determined amount may be programmable. It may also be different from one output port to the next, or from one egress queue 112 to the next for the same interface I or from one sub-queue 113 to the next for the same combination of interface I and egress queue Q. In other embodiments, all the discard probabilities for the same (I,Q) combination can be sent as soon as one of them changes beyond a pre-determined absolute or relative amount. A timer may also be provided in case there is no substantial difference in the discard probability, so that the value is sent to the PAUs 118 at least as often as a pre-determined number of times per second.

Moreover, according to one embodiment of the present invention, a discard probability is independently generated for each combination of output port, egress queue associated with that output port and priority. In

other embodiments, the priority of a packet does not figure into the decision as to whether a packet is discarded or forwarded and hence a single discard probability would be associated with each valid combination of output port and egress queue.

Furthermore, it has been previously mentioned that, in the case where it is decided that the received packet is not to be transmitted, the packet may be discarded from memory altogether or sent to a separate memory store for discarded packets. In other embodiments, packets that are not to be transmitted into the processing fabric can be rerouted along an alternate path.

In still other embodiments, the packets to be discarded may be "marked" as "discardable" but not necessarily discarded unless and until the space they occupy in memory is needed. In this way, if the congestion which led to a packet being "marked" subsides, the packet can be unmarked and may continue on its way to the switch fabric. The characterization of a packet as "marked" may be specified in the packet's header, for example. In yet other embodiments, marked packets may nevertheless be transmitted through the switch fabric but the marking may be used as a signal to the higher layer application that a particular flow must be reduced promptly.

It should also be apparent that although the above description has made reference to a "discard" probability, this need not mean that packets are actually discarded according to such probability. An example is in the case where packets not transmitted to the switch

fabric 102 are stored in a separate memory or logged for future reference. Thus, the term "discard probability" make be regarded as referring to the broader concept of a probability of non-transmission. The decision rendered by a PAU 118 in respect of a received packet is one of transmission or non-transmission, based on the "discard" probability associated with the egress queue for which the packet is destined..

Those skilled in the art should appreciate that in some embodiments of the invention, all or part of the functionality previously described herein with respect to the path acceptance units 118, the discard probability evaluation module 120, the availability processing entity 210, the allocation processing entity 220, the probability processing entity 230 and the extrapolation processing entity 240 may be implemented as pre-programmed hardware or firmware elements (e.g., application specific integrated circuits (ASICs), electrically erasable programmable read-only memories (EEPROMs), etc.), or other related components.

In other embodiments of the invention, all or part of the functionality previously described herein with respect to the path acceptance units 118, the discard probability evaluation module 120, the availability processing entity 210, the allocation processing entity 220, the probability processing entity 230 and the extrapolation processing entity 240 may be implemented as software consisting of a series of program instructions for execution by a digital computer, including a processing unit and a memory connected by a communication bus. Such

memory includes data and the program instructions. The processing unit is adapted to process the data and the program instructions in order to implement the functional blocks described in the specification and for which the operation is depicted in the drawings.

The program instructions could be stored on a medium which is fixed, tangible and readable directly by the computer system, (e.g., removable diskette, CD-ROM, ROM, or fixed disk), or the program instructions could be stored remotely but transmittable to the computer system via a modem or other interface device (e.g., a communications adapter) connected to a network over a transmission medium. The transmission medium may be either a tangible medium (e.g., optical or analog communications lines) or a medium implemented using wireless techniques (e.g., microwave, infrared or other transmission schemes).

Those skilled in the art should further appreciate that the program instructions may be written in a number of programming languages for use with many computer architectures or operating systems. For example, some embodiments may be implemented in a procedural programming language (e.g., "C") or an object oriented programming language (e.g., "C++" or "JAVA").

While specific embodiments of the present invention have been described and illustrated, it will be apparent to those skilled in the art that numerous modifications and variations can be made without departing from the scope of the invention as defined in the appended claims.